DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Tercer Trabajo Práctico

Junio de 2010 — Algoritmos y Estructuras de Datos III

Integrante	LU	Correo electrónico
Bianchi, Mariano	92/08	marianobianchi08@gmail.com
Brusco, Pablo	527/08	pablo.brusco@gmail.com
Di Pietro, Carlos Augusto Lyon	126/08	cdipietro@dc.uba.ar

Facultad de Ciencias Exactas y Naturales

Universidad de Buenos Aires

Ciudad Universitaria - (Pabellón I/Planta Baja)

Intendente Güiraldes 2160 - C1428EGA

Ciudad Autónoma de Buenos Aires - Rep. Argentina

Tel/Fax: (54 11) 4576-3359

http://www.fcen.uba.ar

Índice

1. Ejercicio 1

2. Ejercicio 2

2.1. Introducción

En esta sección se presentara un algoritmo exacto para resolver el problema de encontrar la Clique Máxima en un grafo.

Aun no se conocen algoritmos buenos, es decir, polinomiales con respecto al tamaño de la entrada, para resolver este problema, asi que nos consentraremos en realizar mejoras al algoritmo de fuerza bruta que considera todos los casos.

2.2. Explicación

Un algoritmo de fuerza bruta para resolver el problema de Max-Clique podría simplemente intentar formar el conjunto más grande de nodos, donde ese conjunto sea completo, intentando todas las posibilidades eligiendo todos los conjuntos de un cierto tamaño, luego intentar con un tamaño menor, etc. Probablemente la complejidad de un algoritmo de este estilo sea n^n donde n es la cantidad de nodos del grafo.

Una mejora que surge casi inmediatamente es utilizar la técnica de BackTracking, cuya función principal es intentar podar el arbol implicito de combinaciones posibles.

De todas maneras, implementar solo un BackTracking parece ser poco con respecto a las mejoras que se pueden lograr. A continuación, se explicara el algoritmo implementado con un pseudocodigo y se verá cada una de las mejoras por separado. **Algoritmo Exacto**(G: grafo)

```
1  CliqueMayor = vacia
2  componentesConexas = DetectarComponentesConexas(G)
3  Para cada componente en componentesConexas {
4      heap = crearHeap(G,componentes)
5      Mientras noVacio(heap) ∧ top(heap) ≥ tam(CliqueMayorActual){
6        v = top(heap)
7        Para tamCliqueABuscar desde grado(v)+1 hasta tam(CliqueMayorActual)+1{
8          vecinosFiltrados = filtrarVecinosMenores(v, tamCliqueABuscar-1)
9          Si tam(vecinosFiltrados) +1 ≥ tamCliqueABuscar
10           temp = BuscoCliqueDeTamañoK(tamCliqueABuscar, vecinosFiltrados)
11           Si tam(CliqueMayorActual) < tam(temp)
12             CliqueMayorActual = temp
13       }
14     }
```

Algorithm 1: Pseudocódigo del algoritmo exacto

3. Ejercicio 3

4. Ejercicio 4

5. Anexos